This paper has been accepted for publication at the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Tucson, US, 2025.

FaVoR: Features via Voxel Rendering for Camera Relocalization

Vincenzo Polizzi[1] Marco Cannici[2] Davide Scaramuzza[2] Jonathan Kelly[1]
[1] University of Toronto, [2] University of Zurich
{vincenzo.polizzi,jonathan.kelly}@robotics.utias.utoronto.ca, cannici@ifi.uzh.ch

Abstract

Camera relocalization methods range from dense image alignment to direct camera pose regression from a query image. Among these, sparse feature matching stands out as an efficient, versatile, and generally lightweight approach with numerous applications. However, feature-based methods often struggle with significant viewpoint and appearance changes, leading to matching failures and inaccurate pose estimates. To overcome these limitations, we propose a novel approach that leverages a globally sparse but locally dense 3D representation of 2D features. By tracking and triangulating landmarks over a sequence of frames, we construct a sparse voxel map optimized to render image patch descriptors observed during tracking. Given an initial pose estimate, we first synthesize descriptors from the voxels using volumetric rendering and then perform feature matching to estimate the camera pose. This method enables the generation of descriptors for unseen views, enhancing robustness to viewpoint changes. We evaluate our method on the 7-Scenes and Cambridge Landmarks datasets. Our results show that our approach significantly outperforms existing state-of-the-art feature representation techniques in indoor environments, achieving up to a 39% improvement in median translation error. Additionally, our approach yields comparable results to other methods for outdoor scenes but with lower computational and memory footprints.

Supplementary Material: For additional details, please visit: https://papers.starslab.ca/favor

1. Introduction

Despite extensive research, visual localization remains a significant challenge, particularly under wide viewpoint and appearance changes. Various strategies have been proposed to improve localization performance in difficult scenarios. Sequence-based place recognition, for instance, exploits consistency across frames to identify previously visited locations [27], while geometric constraints derived from point cloud alignment and 3D-3D matching can provide additional, useful information [23, 24, 47]. Ultimately, the effectiveness of many of these strategies depends upon the reliability and robustness of low-level feature matching, which is used extensively to determine the camera pose. Recent efforts to enhance visual localization performance have explored techniques based on neural radiance fields (NeRF) [26]. These approaches build upon dense or semi-dense simultaneous localization and mapping (SLAM) frameworks [9, 26, 33, 41, 52] by either synthesizing frames for feature point extraction or by directly performing photometric alignment. Other techniques exploit neural view synthesis to render a dense descriptor space in order to localize in a known map [21, 28], or to capture the appearance variations of a sparse set of feature points observed from a query viewpoint [14]. While these methods do improve performance, they come with some significant drawbacks. The dense descriptor representation [21, 28] often requires more training time and memory, since a dense and fully optimized radiance field must be learned prior to deployment. Furthermore, works that synthesize sparse descriptors [14] struggle to render descriptors with a large number of channels.

Building on these insights, we introduce FaVoR, a new feature rendering method that exploits a pre-trained neural network to extract features and a sparse voxel-based formulation to encode and render feature descriptors in 3D space. Our scene representation, illustrated in Figure 1, is globally sparse but locally dense, allowing for efficient extraction of view-conditioned 3D point descriptors from any query camera pose. During training, we extract and track keypoints across a sequence of frames, then triangulate 3D landmarks using known camera poses. This approach mirrors existing online localization pipelines [30], which are inherently sequential. Each landmark is represented as a voxel and optimized through volumetric rendering, allowing the associated descriptor to be rendered from novel views. In turn, we are able to carry out low-level feature matching under wide viewpoint changes. Unlike prior techniques that learn dense radiance fields [21, 28], FaVoR encodes sparse landmark descriptors only, offering a favorable tradeoff between descriptor robustness and resource efficiency. FaVoR reduces

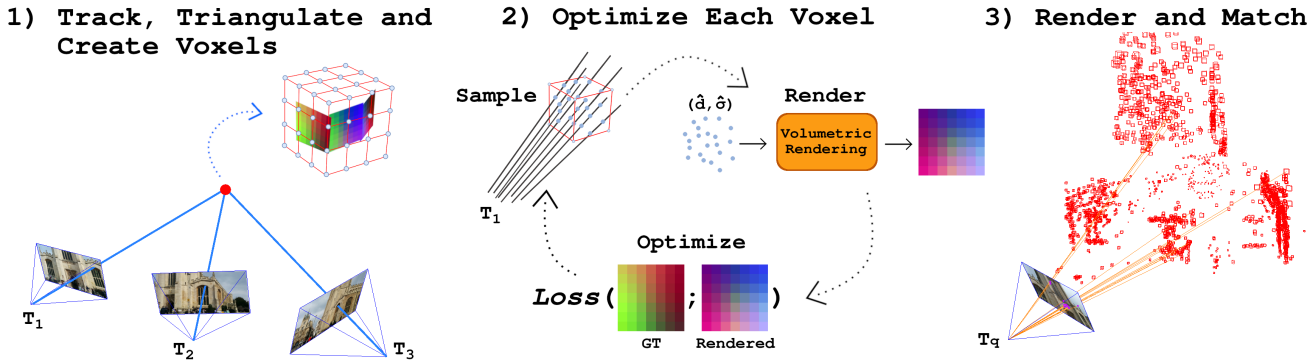


Figure 1. **Schematic representation of the proposed method.** 1) We track and triangulate feature points to create a voxel representation for persistent landmarks, i.e. landmarks observed by a large number of views. After that, 2) the voxels are optimized to render the descriptors patches extracted during the feature tracking. At test time, 3) the voxels are queried to render the descriptors as seen from a given query pose, and the 2D-3D matches between the query image and the landmarks are found and used to perform pose estimation.

the computational burden associated with more dense models, improving scalability in practical applications. Moreover, instead of using a neural network to render descriptors [14, 21, 28], we employ an explicit voxel representation and trilinear interpolation for rendering, accelerating both the training and rendering processes. We extensively evaluate our approach on the 7-Scenes [40] and Cambridge Landmarks [17] datasets. FaVoR significantly outperforms state-of-the-art implicit feature rendering approaches on the indoor 7-Scenes dataset [40], achieving up to a 39% reduction in median translation error, and delivers comparable performance on outdoor scenes.

To summarize, we make the following contributions.

- We present FaVoR, a sparse, voxel-based algorithm to render feature points tracked over a sequence of frames, that does not require learning a dense volumetric representation of the scene.
- We show how to render high-dimensional descriptors, which provide superior view-invariance, and test our descriptor representation using different state-of-the-art feature points extractors with different numbers of channels: 64, 96, 128 [50], and 256 [11].
- We demonstrate using the 7-Scenes [40] and Cambridge Landmarks [17] datasets that our system is capable of rendering descriptors faithfully from unseen views; FaVoR improves camera pose estimation performance compared to other state-of-the-art implicit descriptors rendering techniques [14, 21, 28].
- We release our codebase and pre-trained models to the community.

2. Related Work

We divide traditional visual localization systems into three categories according to the scene representation employed: image-based, structure-based, and hybrid. This section provides an overview of related work in these categories and contrasts traditional methods with newer techniques (collectively called *scene feature renderers*) that synthesize new descriptors based on a query pose.

Image-based methods (IBMs) for visual localization use appearance information only to determine the camera pose for a given query image. A set of known images and associated camera poses is required, which may simply be stored in a database and searched when the query image is fed to the system, or used to train a neural network that learns to regress camera pose. Hence, this category includes absolute [7, 17, 31, 38] and relative [2, 20, 45] pose regression, pose triangulation, and pose interpolation [44] approaches. Most recently, neural radiance fields (NeRF) [26] have been employed to enhance the training of image-based models by providing synthetic views [29]. Prior research has shown that these methods yield limited pose estimation accuracy [37, 51], especially in large environments, due to the approximate nature of pose interpolation.

Structure-based methods (SBMs), on the other hand, use the 3D structure of the scene, typically computed via structure-from-motion (SfM), to recover camera pose through 2D-3D feature point matching. These paradigms can deliver accurate 3D camera pose estimates, particularly when coupled with RANSAC [12] or similar techniques for outlier rejection. The 3D structure of the environment can also be learned end-to-end using neural networks, as demonstrated by SCoRe Forest [40] and DSAC* [5]. Generally, these methods are relatively slow, but recently, ACE [1] has demonstrated fast convergence and better camera pose estimation results than its predecessors. Our work exploits the sparsity of the structure-based representation but uses sequences of frames to track and triangulate landmarks, instead of relying on complete SfM models.

Hybrid methods (HMs) attempt to combine the advantages of the image and structure-based approaches. For each

image in a database, the set of 3D landmarks visible from the associated camera pose can be determined, and matching can be performed between a small set of candidate features and those extracted from the query image. To retrieve the most similar image, full-image descriptors [43] or a vocabulary of visual features [16, 32] can be used. Hybrid methods require significant memory to store the full SfM representation of the scene as well as the landmark descriptors, but they typically achieve the lowest relocalization error in large-scale environments [35]. Some methods aim to reduce memory usage, but at the cost of increased relocalization error [48]. We follow a similar hybrid approach, rendering descriptors only for those landmarks that are likely to be visible from a given camera pose.

Scene feature rendering methods (SFRMs) form a newer class of visual localization techniques that directly render 3D point descriptors from novel views in order to localize. These methods potentially overcome one of the major limitations of traditional feature extractors, that is, the lack of viewpoint invariance. Although learning-based approaches have led to improvements in feature extraction and matching, the network training process still relies on 2D images, and hence the learned descriptors are closely tied to image appearance and viewpoint. To address the problem of viewpoint invariance, FQN [14] learns to model the appearance of feature points observed from different views, but struggles to render high-dimensional descriptors (see [14, Section 6]). Moreau et al. propose CROSSFIRE in [28], a NeRF model that builds a dense scene representation in descriptor space, rather than in the RGB space, allowing for continuous descriptor rendering across different views. Most recently, Liu et al. describe NeRF-loc [21], an approach that leverages the depth information provided by an RGB-D camera to create an accurate NeRF model and then to learn 3D descriptors that can be matched with new views via a multi-headed attention mechanism. Despite the pose estimation accuracy obtained, these methods are limited by training time and memory usage, as they require a full 3D model of the environment. Consequently, they are unsuitable for large-scale and real-time localization tasks. Our approach relies on volumetric rendering but for a sparse set of voxels only, derived from features tracked over a sequence of RGB input images.

3. Methodology

We track a sparse set of landmarks through a sequence of training images. We then use volumetric rendering to optimize a local, dense voxel grid representation of each landmark, rendering the 2D descriptor patches observed during tracking (see Figure 1). At inference time, we iteratively refine the camera estimate by querying the optimized voxel set to determine each landmark's appearance (i.e., feature) based on an initial camera pose guess, as proposed by [14]. The rendered descriptors are then matched with the features in the query image, and these correspondences are used to compute the camera pose by solving the PnP problem [19] within a RANSAC scheme [8, 12]. We refer to this iterative process as Render+PnP-RANSAC. In the following subsections, we describe each step in more detail.

3.1. Landmark Tracking

Consider a sequence of M RGB images, each $H \times W$ pixels in size, $\mathbf{I}_{1,\dots,M} \in \mathbb{R}^{H\times W\times 3}$, and a landmark $\boldsymbol{\ell}_j \in \mathbb{R}^3$ in the world frame. Define $\mathcal{S}_j$ as the set of indices of training images that include $\boldsymbol{\ell}_j$ (i.e., that observe $\boldsymbol{\ell}_j$). For each image $\mathbf{I}_i$, with $i \in \mathcal{S}_j$, there exists a keypoint $\mathbf{k}_{ij} \in \mathbb{R}^2$ that is the projection of $\boldsymbol{\ell}_j$ onto $\mathbf{I}_i$. The camera pose at $\mathbf{I}_i$ is $\mathbf{T}_i \in \mathrm{SE}(3)$ in the world frame. We denote by $\mathcal{F}$ a feature extractor that, given image $\mathbf{I}_i$, provides keypoints and a dense descriptor map $\mathbf{D}_i \in \mathbb{R}^{H\times W\times C}$, where C is the number of descriptor channels. From $\mathbf{D}_i$, we crop patches of size $S \times S$ pixels, $\mathbf{P}_{ij} \in \mathbb{R}^{S\times S\times C}$, centered at each extracted keypoint $\mathbf{k}_{ij}$,

$$\mathbf{P}_{ij} = \mathrm{crop}(\mathbf{D}_i, \mathbf{k}_{ij}, S). \tag{1}$$

For each landmark $\boldsymbol{\ell}_j$, we store a *track* that includes the camera poses $\mathbf{T}_i$ for the sequences of images that include $\boldsymbol{\ell}_j$, the set of keypoints $\mathbf{k}_{ij}$, and the corresponding descriptor patches $\mathbf{P}_{ij}$.

Given a track, we triangulate the landmark position $\boldsymbol{\ell}_j$ in the world frame. An initial estimate of $\boldsymbol{\ell}_j$ is found using the direct linear transform algorithm [15] and refined by minimizing the reprojection error in a Levenberg–Marquardt optimization technique. To account for the presence of outliers, we apply a robust cost in the optimization process [13]. To ensure the numerical stability of the optimization, we used the inverse depth parameterization of the landmark position in the camera frame. The measurement model and cost function are described in detail in our supplementary materials. Our approach is designed to integrate seamlessly into standard front-end pipelines for robot localization, making it suitable for existing online vision-based localization systems like visual-inertial odometry (VIO) [10, 30].

3.2. Voxel Creation

We instantiate a new voxel $\mathcal{V}_j$, centred at $\boldsymbol{\ell}_j$, for each track that is longer than a certain, minimum length, as common localization system do for persistent landmarks selection [30]. The landmark associated with a track must be visible from many poses to provide useful localization information. Each voxel $\mathcal{V}_j$ consists of a grid, containing a number of smaller subvoxels, with a resolution of $R \times R \times R$. Each node (i.e., vertex) of the resulting grid stores a vector of size C, corresponding to the descriptor channels provided by $\mathcal{F}$. To determine the appropriate size of the voxel

(and its subvoxels), we first compute the Euclidean distance l_{ij} from $\mathbf{T}_i$ to the point $\boldsymbol{\ell}_j$, for each patch $\mathbf{P}_{ij}$. Then, given the patch size of $S \times S$ pixels, we estimate the voxel size $s_{\mathcal{V}_j}$ (in metres) as

$$s_{\mathcal{V}_j} = \min_{i \in \mathcal{S}_j} (S \cdot \frac{l_{ij}}{f}), \tag{2}$$

where f is the camera focal length. We choose the minimum voxel size since this is sufficient to capture the information required to render the descriptor associated with the keypoint $\mathbf{k}_{ij}$. Additionally, each voxel $\mathcal{V}_j$ is associated with a density grid of the same resolution and size as the descriptors grid, but with nodes of size one instead of C. The density grid is used in the volumetric rendering process described in Section 3.3.

3.3. Descriptor Learning and Rendering

Once the voxels are created, we can train our system to render the descriptor patches observed along the associated tracks. The training process is similar to the method described in [26], but with notable differences, such as the absence of a multi-layer perceptron (MLP) for view-dependent rendering. For all patches and poses $i \in \mathcal{S}_j$, we trace rays from the camera center $\mathbf{T}_i$ that pass through each element of the patch $\mathbf{P}_{ij}$. Each of the rays $\mathbf{r}$ intersects the voxel grid $\mathcal{V}_j$ and its associated density grid at two points, one closer to the camera, $\mathbf{p}_n$, and one farther away, $\mathbf{p}_f$. We take N samples, $\mathbf{d}_t \in \mathbb{R}^C$ and $\hat{\sigma_t} \in \mathbb{R}$, with $t = 1, \ldots, N$, along the ray between the two intersection points, using trilinear interpolation from both $\mathcal{V}_j$ and the density grid, respectively (see Figure 1).This process follows the volume rendering approach proposed in [25], but instead of rendering an RGB color, we render a feature descriptor,

$$\delta = \frac{||\mathbf{p}_f - \mathbf{p}_n||_2}{N}, \tag{3}$$

$$\hat{\mathbf{d}}_{ij}^{uv} = \sum_{t=1}^{N} T_t \left(1 - \exp\left(-\hat{\sigma_t}\delta\right)\right) \mathbf{d}_t, \tag{4}$$

$$T_t = \prod_{l=1}^{t-1} \exp\left(-\hat{\sigma_l}\delta\right), \tag{5}$$

where $\hat{\mathbf{d}}_{ij}^{uv}$ is the estimated descriptor for landmark $\boldsymbol{\ell}_j$ seen from the pose $\mathbf{T}_i$ at the pixel location (u, v) in patch $\mathbf{P}_{ij}$. To learn the descriptor and density grids, we want to ensure that $\hat{\mathbf{d}}_{ij}^{uv}$, in the descriptor vector space, is as close as possible in norm and direction to the ground truth descriptors $\mathbf{d}_{ij}^{uv} \in \mathbf{P}_{ij}$ extracted by $\mathcal{F}$ for all the $(u, v) \in \{(0,0), (0,1), \ldots, (S,S)\}$ in the patch. We use the mean squared error (MSE) and cosine similarity losses to drive the rendered descriptor's norm and direction as close as possible to the target feature, along with total variation regularization [34] to ensure a smooth representation of the patches $\mathbf{P}_{ij}$. For the density, we use the same entropy loss as described in [42]. In our supplementary material, we provide more details on the losses used. Note that we apply the training process to each voxel independently of all the other voxels, enabling parallelization of the training process.

To render the descriptors from the voxels in the scene observed from a novel view, our system requires an initial guess $\hat{\mathbf{T}}_q$ for the query pose $\mathbf{T}_q$ that we aim to estimate. We assume that this estimate is provided, as is typical in robotics localization systems where a prior pose estimate is available. Given the pose $\hat{\mathbf{T}}_q$ and the set of voxels $\mathcal{V} = \{\mathcal{V}_0, \mathcal{V}_1, \ldots, \mathcal{V}_J\}$ in the scene, we can render the descriptors for all of the landmarks that are visible from a given query pose (this rendering may include points that are occluded because of the lack of depth information). For each $\mathcal{V}_j$, we trace a ray from the query camera pose $\hat{\mathbf{T}}_q$ to the voxel-grid center $\boldsymbol{\ell}_j$ (i.e., the position of the landmarks). Following Equations (3), (4) and (5), we then perform volumetric rendering along the ray to obtain the expected descriptor as seen from $\hat{\mathbf{T}}_q$.

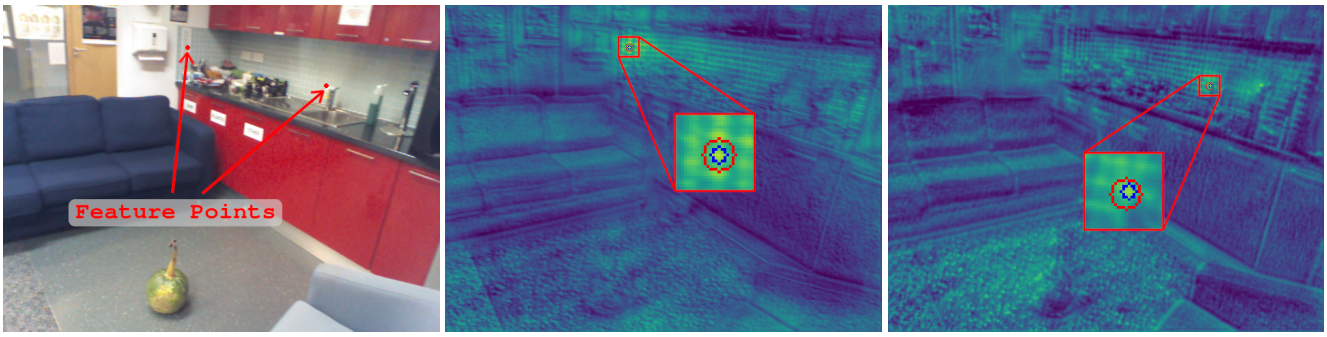
3.4. 2D-3D Matching and Pose Estimation

Once all the descriptors visible from $\hat{\mathbf{T}}_q$ are rendered, we can find 2D-3D correspondences with the query image $\mathbf{I}_q$. A feature extractor generally finds sparse 2D keypoints in the query image and associates each of them with a descriptor extracted from a dense descriptor map given by $\mathcal{F}$ [11, 49]. To match our rendered descriptors to those extracted from the query image, we find the correspondences that provide the highest similarity score, above a certain threshold; that is, we compute a similarity matrix between the two descriptor sets, and after thresholding, we consider only the descriptor pairs with the maximum similarity response. Figure 2 shows the similarity response of a rendered descriptor with the dense descriptor map extracted by the Alike-l [50] feature extractor from an unseen view.

Once all the rendered features are matched and the putative 2D-3D correspondences are available, we use PnP [19] in a RANSAC [8] scheme to determine the camera pose. Our feature representation allows us to render descriptors from novel views, enabling the usage of the iterative Render+PnP-RANSAC refinement procedure. This process highlights the distinctive characteristics of the proposed 3D feature representation. As the estimated query pose $\hat{\mathbf{T}}_q$ converges to $\mathbf{T}_q$, the rendered descriptors increasingly match the query image descriptors' appearance, resulting in a higher number of correspondences (see Section 4).

4. Results

We evaluate our system for camera relocalization in both indoor and outdoor environments using two well-established benchmarks: the 7-Scenes [40] and Cambridge Landmarks [17] datasets. We provide implementation de-



(a) (b) (c)

Figure 2. **Visualization of similarity response.** We render a feature tracked during training using the Alike-l descriptor from an unseen view. On the left, a) displays the ground truth positions of the rendered feature points, obtained by projecting the triangulated landmarks on the camera plane, in red. At the same time, b) and c) show the similarity response between the rendered features and the target image dense descriptor map. The yellow colour indicates a strong response, concentrated around the feature positions shown in a), demonstrating the effectiveness of our descriptor rendering approach. The small circle in blue is the circle centre at the strongest response, the red circle is centred at the project landmark position. The top three response peaks per keypoint are 0.897, 0.868, and 0.836 for b) and 0.856, 0.786, and 0.737 for c) obtained by performing non-maxima suppression with a ray of three pixels, our method effectively renders descriptors that are distinctive in the image.

Dataset	Feature Extractor	Median Pose Error (cm, deg) ↓			Avg. Inliers per Image ↑		
		Iter 1	Iter 2	Iter 3	Iter 1	Iter 2	Iter 3
7-Scenes	FaVoR $_{\text{Alike-l}}$	1.5, 0.4	1.4, 0.4	**1.3, 0.4**	49	54	54
	FaVoR $_{\text{SP}}$	1.8, 0.5	1.6, 0.4	**1.5, 0.4**	73	81	81
Cambridge	FaVoR $_{\text{Alike-l}}$	18.0, 0.3	16.8, 0.3	**15.6, 0.3**	205	222	221
	FaVoR $_{\text{SP}}$	20.8, 0.4	19.6, 0.3	**18.2, 0.3**	200	218	218

Table 1. **Comparison of different feature extractor networks**. We compare different feature extractors for pose estimation using FaVoR rendering. Reported are the median translation and rotation error and the number of 2D-3D match inliers per image. The subcolumns indicate results obtained after 1 to 3 PnP-RANSAC scheme iterations. In **bold** are the best overall pose estimate results.

tails and information about our baseline comparisons in Section 4.1. In Section 4.2, we evaluate our method against existing state-of-the-art (SOTA) relocalization systems that use different scene representation paradigms. We report on the memory footprint and discuss the computational requirements of FaVoR compared to other systems in Section 4.3. Lastly, in Section 4.4, we examine the rendering capabilities of FaVoR as a function of the viewing angle.

4.1. Implementation Details and Baselines

Our implementation is based on PyTorch, building on the framework provided by [42]. We use Alike [50] and SuperPoint [11] as feature extractors. We test descriptors with 64 (Alike-t), 94 (Alike-s), 128 (Alike-n), 128 (Alike-l), and 256 (SuperPoint) channels. Training is carried out for 2,000 epochs per voxel (landmark) using 1,024 rays per epoch, with the Adam optimizer [18] and a variable learning rate per subvoxel as in [42]. We examine the trade-off between model size, rendering quality, expressed by the signal-to-noise ratio (PSNR), and grid resolution (details in the supplementary material), and chose a grid resolution of $3 \times 3 \times 3$ to balance memory usage and rendering performance. Training a voxel on a 7×7 pixel descriptor patch takes about 10 seconds on an NVIDIA GeForce RTX 4060 laptop GPU, rendering a single descriptor takes about 1 ms. However, FaVoR is not currently optimized for fast runtime operations. Since the voxels are independent of each other, the entire training and inference pipeline can be parallelized to significantly improve performance.

We compare FaVoR with similar SFR methods that render descriptors from unseen views: FQN [14], CROSSFIRE [28], and NeRF-loc [21], which closely align with our work in terms of scope, despite the different methodologies and requirements (see Section 2). FQN [14] and CROSSFIRE [28] rely on large sets of points, using SfM and dense feature matching, respectively. NeRF-loc [21] matches each query image with 1,024 3D points. In contrast, our approach uses a smaller set of points. Additionally, while [21, 28] require a fully trained neural rendering model to work, FaVoR needs only a sequence of frames to train the sparse voxel representation of the landmarks. Moreover, FaVoR works with out-of-the-box feature extrac-

Method	Prior Err.	Iter 1	Iter 2	Iter 3
Retrieval	21.9, 12.13	0.8, 0.21	0.7, 0.19	0.7, 0.18
Constant	147.6, 29.94	1.0, 0.28	0.7, 0.19	0.7, 0.18

Table 2. **Different pose initialization priors.** We report the median localization errors (cm, deg) at different iterations of the Render+PnP-RANSAC scheme on the chess scene of the 7-Scenes dataset. The pose initialization algorithms used are DenseVLAD for *Retrieval* and the first pose of the training set for *Constant*.

tion methods, without requiring a custom, scene-dependent network as in [28]. By training FaVoR using descriptors with different numbers of channels, we demonstrate that our method is scalable to different descriptor sizes, unlike [14]. Also, while [14] iterates the Render+PnP-RANSAC multiple times, we use only three iterations, similarly to [28]. For comprehensiveness, we report the localization results of image-based [6, 17, 29, 39] and structure-based [3, 5, 46] camera relocalization methods, as well as hybrid methods (see Section 2 for more details). For the HM category, we report the results from HLoc [35], based on SuperPoint [11] (SP) and SuperGlue [36] (SG), which uses a SfM point cloud for relocalization. We also include results for SceneSqueezer [48] which requires less memory but at the cost of greater localization error.

4.2. Relocalization Evaluation

Following the setup described in Section 4.1, we observe in Table 1 an increase in the number of inliers, accompanied by a decrease in the rotation and translation errors as the camera pose approaches the (true) pose. This observation, suggests that FaVoR faithfully produces descriptors from poses seen during training as well as for new views.

Given a query test image, we use DenseVLAD [43] to retrieve the pose $\hat{\mathbf{T}}_q$ of the most similar image observed during training. This approach is also consistent with the methods used by [14] and [28]. To evaluate the impact of poor initialization on localization, we used the first pose per scene from the 7-Scenes dataset [40] as a uniform prior for all images, instead of DenseVLAD [43] pose priors.This simulates increasing noise as the query pose moves further from the initial pose. As shown in Table 2 (more results in the supplementary), despite imprecise priors, the Render+PnP-RANSAC iterations reliably yield accurate localization, demonstrating the robustness of our method to initial pose variations.

Indoor: 7-Scenes. The 7-Scenes [40] dataset includes a sequence of frames recorded with a Kinect RGB-D camera, of seven different indoor environments. The data presents motion blur, small baselines between the frames, and occlusion, all typical elements for real-life indoor robotics applications. We use the full image size of 640×480 pixels for training and testing. For evaluation, we used the poses provided by [4] as ground truth. In Table 3, we report the median translation and rotation errors (in centimetres and degrees) for state-of-the-art (SOTA) pose relocalization methods, divided into categories according to the data representation paradigm. Our system outperforms the SOTA SFR methods by 39% and 69% for the translational and rotational errors, respectively. FaVoR also yields better results than the SBMs, IBMs, and HLoc [35] approaches. In particular, for the indoor dataset, we report the results from the implementation repository of HLoc [35], which are obtained using the depth maps provided by [5]. We limited the number of landmarks for the indoor environments to 1,500.

Outdoor: Cambridge Landmarks. The Cambridge Landmarks [17] dataset includes images from large outdoor scenes that are 875 to 5,600 m^2 in size, with significant view and appearance changes. We reduce the size of the images to 512×288 pixels for training and testing, since Alike [50] is faster when using lower-resolutions. For testing on the outdoor data set, we optimized 5,000 voxels for the Shop and Hospital scenes, and 10,000 for the others, based on scene size. As shown in Table 4, similar to the indoor environment results, our system outperforms the SB and IB methods, with the hybrid methods achieving the best performance in large scenarios. For the evaluation of HLoc [35], we report the results from LIMAP [22], which, unlike the original HLoc, account for the camera lens distortion model when reprojecting the landmarks, providing the best results for outdoor environments. FaVoR produces the second-best camera pose estimation results in the SFR category. In particular, FaVoR yields better results than FQN [14] and CROSSFIRE [28] for all of the scenarios, while NeRF-loc [21] consistently provides an error of 1-3 centimeters lower than FaVoR. One of the main reasons for the reduced performance relative to NeRF-loc [21] is the inaccurate estimate of the position of the landmarks computed during the triangulation step. Unlike NeRF-loc [21], which utilizes a fully trained NeRF model and RGB-D images for accurate 3D scene rendering, we adopt a sparse visual localization front-end that triangulates landmarks using tracks, which could introduce inaccuracies. The pose estimation errors observed with FaVoR in the outdoor dataset may result from insufficient viewpoint variation, which makes accurate triangulation challenging given the large scene depth. However, despite the lower accuracy of FaVoR for outdoor scenes, our method with Alike, as analyzed in Section 4.3, uses less memory than [21] and does not require a complete neural rendering of the scene.

4.3. Memory Usage and Training Complexity

In Table 5, we compare the checkpoint memory requirements of FaVoR for both datasets against other methods. FaVoR uses more memory than FQN [14] as our method represents each landmark independently. However, this en-

Category	Method	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Average
		(cm, deg)							
IBMs	PoseNet17 [17]	13, 4.5	27, 11.3	17, 13.0	19, 5.6	26, 4.8	23, 5.4	35, 12.4	22.9, 8.1
	MapNet [6]	8, 3.3	27, 11.7	18, 13.3	17, 5.2	22, 4.0	23, 4.9	30, 12.1	20.7, 7.8
	PAEs [39]	12, 5.0	24, 9.3	14, 12.5	19, 5.8	18, 4.9	18, 6.2	25, 8.7	18.6, 7.5
	LENS [29]	3, 1.3	10, 3.7	7, 5.8	7, 1.9	8, 2.2	9, 2.2	14, 3.6	8.3, 3.0
HM	$\text{HLoc}^{\text{RGB-D}}_{\text{SP+SG}}$ [35]	2, 0.8	2. 0.8	1, 0.8	3, 0.8	4, 1.1	3, 1.1	4, 1.2	2.7, 0.9
SBMs	SC-WLS [46]	3, 0.8	5, 1.1	3, 1.9	6, 0.9	8, 1.3	9, 1.4	12, 2.8	6.6, 1.5
	DSAC* (RGB) [5]	2, 1.1	2, 1.2	1, 1.8	3, 1.2	4, 1.3	4, 1.7	3, 1.2	2.7, 1.4
	ACE [3]	2, 1.1	2, 1.8	2, 1.1	3, 1.4	3, 1.3	3, 1.3	3, 1.2	2.6, 1.3
SFRMs	FQN [14]	4, 1.3	5, 1.8	4, 2.4	10, 3.0	9, 2.5	16, 4.4	140, 34.7	27.4, 7.4
	CROSSFIRE [28]	1, 0.4	5, 1.9	3, 2.3	5, 1.6	3, 0.8	2, 0.8	12, 1.9	4.4, 1.4
	NeRF-loc [21]	2, 1.1	2, 1.1	1, 1.9	2, 1.1	3, 1.3	3, 1.5	3, 1.3	2.3, 1.3
	(Ours) FaVoR $_{\text{Alike-t}}$	1, 0.3	1, 0.5	1, 0.4	2, 0.6	2, 0.4	1, 0.3	4, 1.1	1.7, 0.5
	(Ours) FaVoR $_{\text{Alike-s}}$	1, 0.2	2, 0.6	1, 0.4	2, 0.4	1, 0.3	4, 0.9	5, 1.5	2.3, 0.6
	(Ours) FaVoR $_{\text{Alike-n}}$	1, 0.2	1, 0.4	1, 0.6	2, 0.4	1, 0.3	1, 0.3	6, 1.6	1.9, 0.5
	(Ours) FaVoR $_{\text{Alike-l}}$	1, 0.2	1, 0.3	1, 0.4	2, 0.4	1, 0.3	1, 0.2	3, 0.8	**1.4, 0.4**
	(Ours) FaVoR $_{\text{SP}}$	1, 0.2	1, 0.4	1, 0.3	2, 0.4	1, 0.3	1, 0.2	4, 1.0	1.6, 0.4

Table 3. **6-DoF median localization errors on the 7-Scenes dataset [40]**. Comparison of visual localization methods. The overall top three results are shown in **bold**, underline, and double-undeline.

Category	Method	College	Court	Hospital	Shop	Church	Average	Average w/o Court
		(cm, deg)						
IBMs	PoseNet [17]	88, 1.0	683, 3.5	88, 3.8	157, 3.3	320, 3.3	267, 3.0	163, 2.9
	MapNet [6]	107, 1.9	785, 3.8	149, 4.2	200, 4.5	194, 3.9	287, 3.7	163, 3.6
	PAEs [39]	90, 1.5	-	207, 2.6	99, 3.9	164, 4.2	-	140, 3.1
	LENS [29]	33, 0.5	-	44, 0.9	27, 1.6	53, 1.6	-	39, 1.2
HM	$\text{HLoc}_{\text{SP+SG}}$ [35]	6, 0.1	10, 0.1	13, 0.2	3, 0.1	4, 0.1	**7, 0.1**	**7, 0.1**
	SceneSqueezer [48]	27, 0.4	-	37, 0.5	11, 0.4	15, 0.4	-	23, 0.4
SBMs	SC-WLS [46]	14, 0.6	164, 0.9	42, 1.7	11, 0.7	39, 1.3	54, 0.7	27, 1.1
	DSAC* (RGB) [5]	18, 0.3	34, 0.2	21, 0.4	5, 0.3	15, 0.6	19, 0.3	15, 0.4
	ACE [3]	28, 0.4	42, 0.2	31, 0.6	5, 0.3	19, 0.6	25, 0.4	21, 0.5
SFRMs	FQN-MN [14]	28, 0.4	4253, 39.2	54, 0.8	13, 0.6	58, 2.0	881, 8.6	38, 1.0
	CROSSFIRE [28]	47, 0.7	-	43, 0.7	20, 1.2	39, 1.4	-	37, 1.0
	NeRF-loc [21]	11, 0.2	25, 0.1	18, 0.4	4, 0.2	7, 0.2	13, 0.2	10, 0.3
	(Ours) FaVoR $_{\text{Alike-t}}$	17, 0.3	29, 0.1	20, 0.4	5, 0.3	11, 0.4	16, 0.3	13, 0.4
	(Ours) FaVoR $_{\text{Alike-s}}$	16, 0.2	32, 0.2	21, 0.4	6, 0.3	11, 0.4	17, 0.3	14, 0.4
	(Ours) FaVoR $_{\text{Alike-n}}$	18, 0.3	32, 0.2	21, 0.4	5, 0.2	11, 0.3	17, 0.3	14, 0.3
	(Ours) FaVoR $_{\text{Alike-l}}$	15, 0.2	27, 0.1	19, 0.4	5, 0.3	10, 0.3	15, 0.3	12, 0.3
	(Ours) FaVoR $_{\text{SP}}$	18, 0.3	29, 0.2	27, 0.5	5, 0.3	11, 0.4	18, 0.3	15, 0.4

Table 4. **6-DoF median localization errors on the Cambridge dataset [17]**. Comparison of visual localization methods. The overall top three results are shown in **bold**, underline, and double-undeline. Some NeRF-based methods fail when trained on the Great Court scene due to the poor image quality, whereas we can still track and train our voxel-based representation.

ables our representation to scale effectively, rendering high-dimensional descriptors in large scenes and consistently outperforming FQN [14]. On the other hand, FaVoR uses less than half the memory required by CROSSFIRE [28]. However, for large outdoor scenes, our method does require more memory. Despite the larger memory footprint, FaVoR

Model	7-Scenes [40]	Cambridge [17]
Posenet [17]	50MB	50MB
DSAC* [5]	28MB	28MB
ACE [3]	4MB	4MB
$\text{HLoc}_{\text{SP+SG}}$ [35] (features only)	$>$ 4.2 GB	$>$ 2.5 GB
FQN [14]	2MB	2MB
CROSSFIRE [28]	50MB	50MB
NeRF-loc [21] (backbone only)	$>$ 129 MB	$>$ 129 MB
Ours (Alike-t)	13MB	83MB
Ours (Alike-l)	19MB	128MB
Ours (SuperPoint)	32MB	216MB

Table 5. **Scene representation size comparison**. Comparison between the size of the largest checkpoints obtained with FaVoR. The Chess [40] and King's College [17] scenes result in larger checkpoints than the other scenes. For HLoc [35], we report the memory used to store the features only in the two scenes.

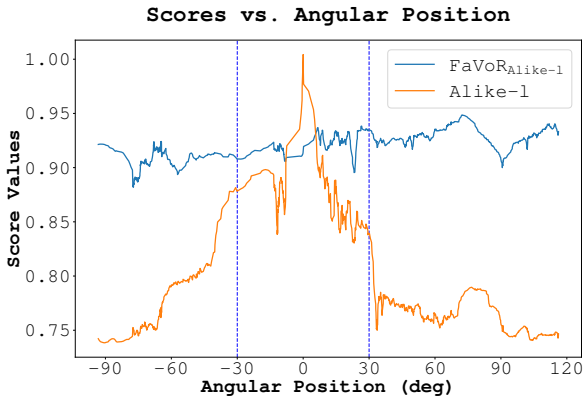


Figure 3. **Median similarity score versus viewing angle.** In blue is the smoothed median score for FaVoR $_{\text{Alike-l}}$ obtained by convolving the descriptors rendered at different view angles with the corresponding dense descriptors map of each query image. In orange is the smoothed median score of Alike-l features extracted from the starting image (at angle 0 deg) convolved with the subsequent images in the test sequence.

delivers consistently better localization performance than CROSSFIRE [28]. FaVoR $_{\text{Alike-l}}$ also requires less memory compared to NeRF-loc [21] for indoor and outdoor scenes.

Our approach also performs favorably compared to hybrid methods. For instance, FaVoR uses over 200 times less memory than HLoc [35] with SP+SG [11, 36] in indoor scenes. Methods like SceneSqueezer [48] significantly reduce the memory requirements (0.3 MB) for HBs, but at the cost of increased localization error. Image-based methods generally require less memory than structure-based ones but with the caveat that they provide less accurate pose estimates in general. Advantages of our method include its ability to train a voxel from feature tracks, making it suitable for robotics localization pipelines.

4.4. Rendering Capabilities

When extracting feature points from an image (i.e., keypoints and associated descriptors), the descriptors must be largely view-invariant to ensure robust matching from different viewpoints. To evaluate the view-invariance of feature descriptors, we extract dense descriptor maps from a sequence of images taken from different angles. For each image, we use Alike-l [50] to extract features from a target image and compute similarity scores between each feature and the dense maps. The same process is applied to descriptors rendered by FaVoR, but using the ground truth pose for rendering.

Figure 3 shows the median values of the top thirty similarity scores per image at different query angles, for Alike-l and FaVoR trained to render Alike-l features. FaVoR maintains an almost constant value, indicating that the descriptors are rendered faithfully from unseen views close to the ones observed during training. In contrast, there is a noticeable degradation in the median similarity scores when attempting to match features directly extracted by Alike-l, with a significant drop occurring beyond ± 30 degrees. Figure 3 demonstrates the effectiveness of FaVoR in maintaining descriptor rendering fidelity across varying viewpoints.

5. Conclusion

In this paper, we presented FaVoR, a novel, globally sparse and locally dense vision-based camera relocalization method. Our approach involves tracking features over a sequence of RGB images with known poses and learning a 3D descriptor representation that models variations in appearance according to viewpoint. Our method is also suitable for integration with existing track-based localization systems for robotics.

There are several directions for future work. The voxels in FaVoR are independent of each other, allowing for versatile training approaches. Although our current implementation optimizes each voxel sequentially, parallelizing this process could significantly speed up training, potentially approaching the speed of training a single voxel. We plan to implement these parallelization techniques in the future to enhance efficiency. To increase robustness against occlusions, additional modifications could involve utilizing full-image descriptors for landmark retrieval.

References

[1] Ignacio Alzugaray and Margarita Chli. ACE: An efficient asynchronous corner tracker for event cameras. In *3D Vision (3DV)*, pages 653–661, 2018. 2

[2] Vassileios Balntas, Shuda Li, and Victor Prisacariu. Relocnet: Continuous metric learning relocalisation using neural nets. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 751–767, 2018. 2

[3] Eric Brachmann, Tommaso Cavallari, and Victor Adrian Prisacariu. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 5044–5053, 2023. 6, 7, 8

[4] Eric Brachmann, Martin Humenberger, Carsten Rother, and Torsten Sattler. On the Limits of Pseudo Ground Truth in Visual Camera Re-Localization. In *Int. Conf. Comput. Vis. (ICCV)*, 2021. 6

[5] Eric Brachmann and Carsten Rother. Visual camera relocalization from rgb and rgb-d images using dsac. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(9):5847–5865, 2021. 2, 6, 7, 8

[6] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-aware learning of maps for camera localization. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 2616–2625, 2018. 6, 7

[7] Shuai Chen, Zirui Wang, and Victor Prisacariu. Direct-posenet: Absolute pose regression with photometric consistency. In *Int. Conf. Comput. Vis. (ICCV)*, pages 1175–1185. IEEE, 2021. 2

[8] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized ransac. In *Pattern Recog.*, pages 236–243. Springer, 2003. 3, 4

[9] Chi-Ming Chung, Yang-Che Tseng, Ya-Ching Hsu, Xiang-Qian Shi, Yun-Hung Hua, Jia-Fong Yeh, Wen-Chin Chen, Yi-Ting Chen, and Winston H Hsu. Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 9400–9406. IEEE, 2023. 1

[10] Jeff Delaune, David S. Bayard, and Roland Brockers. Range-visual-inertial odometry: Scale observability without excitation. *IEEE Robot. Autom. Lett.*, 6(2):2421–2428, 2021. 3, 13

[11] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, pages 224–236, 2018. 2, 4, 5, 6, 8, 11

[12] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 2, 3

[13] Donald Geman and Stuart Geman. Bayesian image analysis. In *IEEE Trans. Syst. Man, Cybern. A, Syst., Humans*, pages 301–319. Springer, 1986. 3, 14

[14] Hugo Germain, Daniel DeTone, Geoffrey Pascoe, Tanner Schmidt, David Novotny, Richard Newcombe, Chris Sweeney, Richard Szeliski, and Vasileios Balntas. Feature query networks: Neural surface description for camera pose refinement. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 5071–5081, 2022. 1, 2, 3, 5, 6, 7, 8

[15] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 3

[16] Zhang Jiachen, Dahang Ai, Yi Xiang, Xin Chang, Yi Wang, and Xiaodong Chen. Bag-of-words based loop-closure detection in visual SLAM. In Xiao-Cong Yuan, Kebin Shi, and Michael G. Somekh, editors, *Proc. Advanced Maui Optical and Space Surveillance Technol. Conf. (AMOS)*, volume 10816, page 1081618. International Society for Optics and Photonics, SPIE, 2018. 3

[17] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Int. Conf. Comput. Vis. (ICCV)*, pages 2938–2946, 2015. 2, 4, 6, 7, 8, 10, 11, 12, 20

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 5

[19] Zuzana Kukelova, Jan Heller, and Andrew Fitzgibbon. Efficient intersection of three quadrics and applications in computer vision. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 1799–1808, 2016. 3, 4

[20] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera relocalization by computing pairwise relative poses using convolutional neural network. In *Int. Conf. Comput. Vis. Workshops (ICCVW)*, pages 929–938, 2017. 2

[21] Jianlin Liu, Qiang Nie, Yong Liu, and Chengjie Wang. Nerf-loc: Visual localization with conditional neural radiance field. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 9385–9392. IEEE, 2023. 1, 2, 3, 5, 6, 7, 8

[22] Shaohui Liu, Yifan Yu, Rémi Pautrat, Marc Pollefeys, and Viktor Larsson. 3d line mapping revisited. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2023. 6

[23] Fabiola Maffra, Lucas Teixeira, Zetao Chen, and Margarita Chli. Loop-closure detection in urban scenes for autonomous robot navigation. In *Int. Conf. Comput. Vis. (ICCV)*, pages 356–364, 2017. 1

[24] Fabiola Maffra, Lucas Teixeira, Zetao Chen, and Margarita Chli. Real-time wide-baseline place recognition using depth completion. *IEEE Robot. Autom. Lett.*, 4(2):1525–1532, 2019. 1

[25] Nelson Max. Optical models for direct volume rendering. *Found. and Trends in Comp. Graph. and Vis.*, 1(2):99–108, 1995. 4

[26] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65(1):99–106, 2021. 1, 2, 4

[27] Michael J Milford and Gordon F Wyeth. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1643–1649, 2012. 1

[28] A. Moreau, N. Piasco, M. Bennehar, D. Tsishkou, B. Stanciulescu, and A. de La Fortelle. Crossfire: Camera relocalization on self-supervised features from an implicit representation. In *Int. Conf. Comput. Vis. (ICCV)*, pages 252–262, Los Alamitos, CA, USA, oct 2023. IEEE Computer Society. 1, 2, 3, 5, 6, 7, 8

[29] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Lens: Localization enhanced by nerf synthesis. In *Conf. on Robotics Learning (CoRL)*, volume 164 of *Proceedings of Machine Learning Research*, pages 1347–1356. PMLR, 2022. 2, 6, 7

[30] Anastasios I. Mourikis and Stergios I. Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3565–3572, 2007. 1, 3

[31] Tayyab Naseer and Wolfram Burgard. Deep regression for monocular camera-based 6-dof global localization in outdoor environments. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, pages 1525–1530. IEEE, 2017. 2

[32] Wisam A. Qader, Musa M. Ameen, and Bilal I. Ahmed. An overview of bag of words;importance, implementation, applications, and challenges. In *IEEE Int. Conf. Neural Netw.*, pages 200–204, 2019. 3

[33] Antoni Rosinol, John J Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, pages 3437–3444. IEEE, 2023. 1

[34] Leonid I Rudin and Stanley Osher. Total variation based image restoration with free local constraints. In *IEEE Int. Conf. Image Process. (ICIP)*, volume 1, pages 31–35. IEEE, 1994. 4

[35] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019. 3, 6, 7, 8

[36] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2020. 6, 8

[37] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019. 2

[38] Yoli Shavit, Ron Ferens, and Yosi Keller. Learning multi-scene absolute pose regression with transformers. In *Int. Conf. Comput. Vis. (ICCV)*, pages 2733–2742, 2021. 2

[39] Yoli Shavit and Yosi Keller. Camera pose auto-encoders for improving pose regression. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 140–157. Springer, 2022. 6, 7

[40] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew W. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 2930–2937, 2013. 2, 4, 6, 7, 8, 11, 12, 14, 15, 18

[41] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. imap: Implicit mapping and positioning in real-time. In *Int. Conf. Comput. Vis. (ICCV)*, pages 6229–6238, October 2021. 1

[42] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2022. 4, 5, 12, 13

[43] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 1808–1817, 2015. 3, 6, 14

[44] Akihiko Torii, Josef Sivic, and Tomas Pajdla. Visual localization by linear combination of image descriptors. In *Int. Conf. Comput. Vis. (ICCV)*, pages 102–109, 2011. 2

[45] Mehmet Ozgur Turkoglu, Eric Brachmann, Konrad Schindler, Gabriel J Brostow, and Aron Monszpart. Visual camera re-localization using graph neural networks and relative pose supervision. In *Int. Conf. Comput. Vis. (ICCV)*, pages 145–155. IEEE, 2021. 2

[46] Xin Wu, Hao Zhao, Shunkai Li, Yingdian Cao, and Hongbin Zha. Sc-wls: Towards interpretable feed-forward camera re-localization. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 585–601. Springer, 2022. 6, 7

[47] Heng Yang, Jingnan Shi, and Luca Carlone. Teaser: Fast and certifiable point cloud registration. *IEEE Trans. Robot.*, 37(2):314–333, 2021. 1

[48] Luwei Yang, Rakesh Shrestha, Wenbo Li, Shuaicheng Liu, Guofeng Zhang, Zhaopeng Cui, and Ping Tan. Scenesqueezer: Learning to compress scene for camera relocalization. In *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 8259–8268, June 2022. 3, 6, 7, 8

[49] Xiaoming Zhao, Xingming Wu, Weihai Chen, Peter CY Chen, Qingsong Xu, and Zhengguo Li. Aliked: A lighter keypoint and descriptor extraction network via deformable transformation. *IEEE Trans. on Instr. and Meas.*, 2023. 4

[50] Xiaoming Zhao, Xingming Wu, Jinyu Miao, Weihai Chen, Peter C. Y. Chen, and Zhengguo Li. Alike: Accurate and lightweight keypoint detection and descriptor extraction. *IEEE Trans. Robot.*, March 2022. 2, 4, 5, 6, 8, 11, 12, 13, 14

[51] Qunjie Zhou, Torsten Sattler, Marc Pollefeys, and Laura Leal-Taixe. To learn or not to learn: Visual localization from essential matrices. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3319–3326. IEEE, 2020. 2

[52] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. In *Int. Conf. Comput. Vis. (ICCV)*, pages 42–52. IEEE, 2024. 1

Supplementary Material

We report on the results obtained by FaVoR at various iterations of the PnP-RANSAC scheme in in Appendix A. In Appendix B, we discuss the tradeoff between the voxel resolution and both the rendering capabilities and matching performance of our system. We also report the similarity responses for the Cambridge Landmarks [17] dataset, discussing the evidence of a lack of accuracy in the landmark triangulation in Appendix C. Finally, we provide more details on our training losses and our landmark triangulation method in Appendices D and E, respectively.

A. Extended Analysis of FaVoR Performance and Error Computation

In this section, we report the pose estimation errors obtained with different feature extractors at various iterations of the iterative PnP-RANSAC scheme. Specifically, we report the values for Alike-t, Alike-s, Alike-n, Alike-l [50] and SuperPoint [11] with 64, 94, 128, 128, and 256 channels descriptors, respectively.

Table 7 for the 7-Scenes [40] and Table 8 for the Cambrdige [17] datasets give the median pose estimate at the 1st, 2nd, and 3rd iterations of PnP-RANSAC, and the respective average number of inlier points per image (used to compute the pose estimate). The tables also report the success rates of the PnP-RANSAC iterative scheme at the various iterations, i.e., the ratio between the number of successful estimates and the total number of queries. The data shows a clear trend. Namely, the average number of inliers per image increases as the estimated camera pose converges towards the true query image pose (i.e., as the pose estimate error decreases). Furthermore, although there is a difference in matching performance between the various Alike networks (as reported in the Alike [50] manuscript), our descriptor representation effectively 'flattens' these differences, enabling robust matching despite view changes.

B. PSNR versus Voxel Resolution

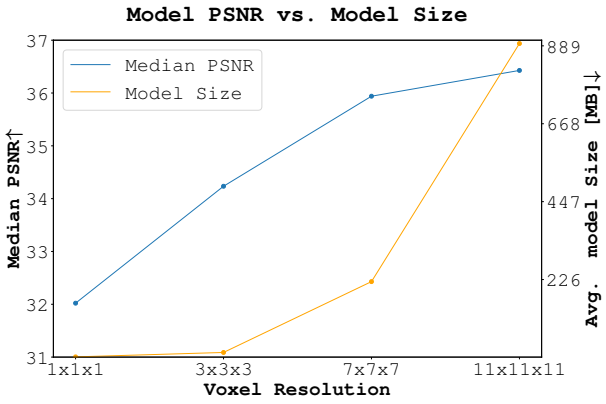


Figure 4. **PSNR and model size versus grid resolution.** We report the median peak signal-to-noise ratio (PSNR) and the average checkpoint size for FaVoR $_{\text{Alike-l}}$ at different grid resolutions of the voxel representation.

The number of sub-voxels in each voxel representing a landmark determines the grid resolution, $R \times R \times R$. The grid resolution directly impacts the rendering quality of the descriptor patches, increasing or decreasing the peak signal-to-noise ratio (PSNR) values. The PSNR calculation

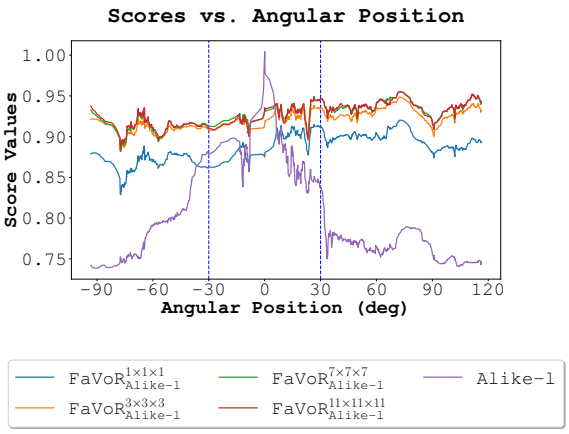


Figure 5. **Similarity response scores versus grid resolution at different view angles.** We compare the different grid resolutions' capacity to provide high score similarity score results at different view angles. Higher scores lead to better matching meaning that the rendered descriptors properly match the appearance of the ones extracted by Alike-l [50].

is given by

$$\text{PSNR} = 10 \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right), \tag{6}$$

where MAX = 2 is the maximum span of the descriptor values, i.e, in the range (-1, 1), and MSE is the mean squared error between the ground-truth patch and the rendered patch. A grid resolution of $R \times R \times R$ implies that the grid contains $R \cdot R \cdot R$ nodes, where each node (vertex) encodes C channels (equal to the number of channels of the descriptor). The chosen grid resolution impacts the overall model size.

The plot in Figure 4 shows the median PSNR values at different grid resolutions and the corresponding model size on the chess scene of the 7-Scenes dataset [40]. The graph shows that beyond a certain grid resolution, the improvement in terms of PSNR is decreasing while, in contrast, the model size grows exponentially. Therefore, as a tradeoff between model size and *good* rendering capabilities of our representation, we choose a grid resolution of $3 \times 3 \times 3$ for our model. Also, this resolution choice is supported by the median score values, obtained as described in the manuscript in Section 4.4, reported in Figure 5. Figure 5 shows the median score values obtained in the chess scene of the 7-Scenes dataset [40] with Alike-l [50], at different grid resolutions. We note that grid resolutions greater than $1 \times 1 \times 1$ yield a similar score response; hence, we choose the lower resolution to save on memory.

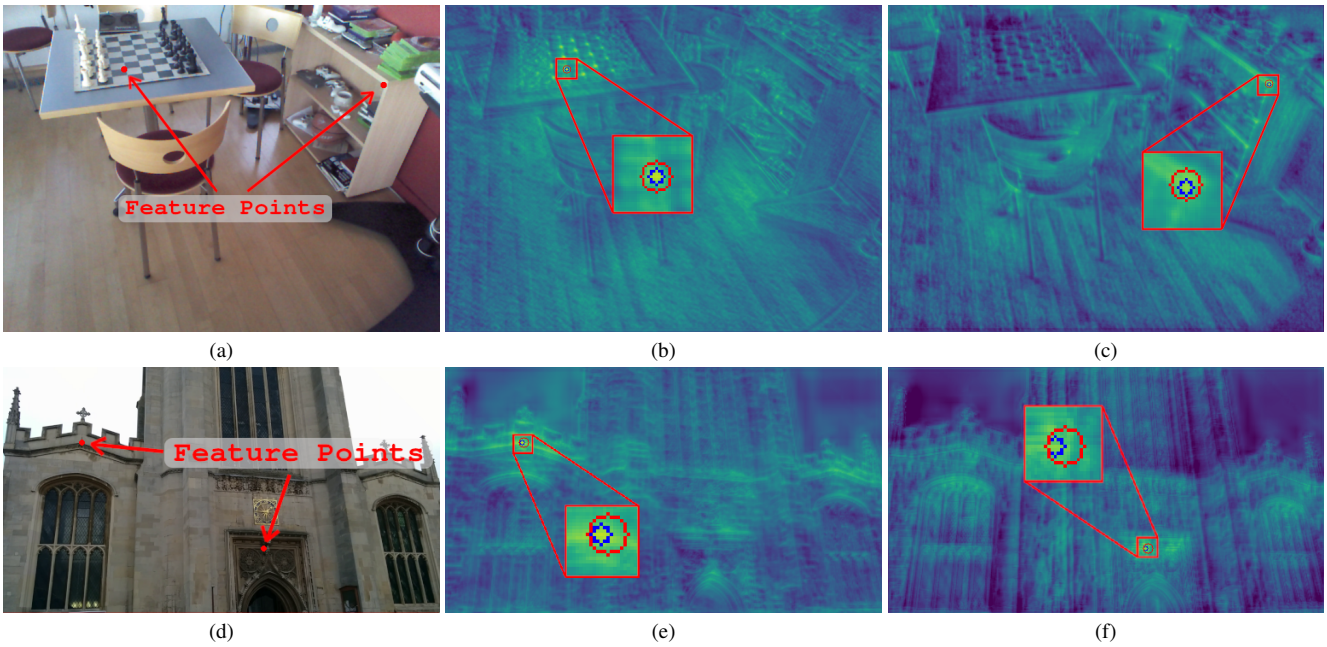


Figure 6. **Visualization of similarity response.** We render a feature tracked during training using the Alike-l descriptor from an unseen view on the 7-Scenes [40] and the Cambridge Landmarks [17] datasets. On the left, a) and d) display the ground truth positions of the rendered feature points, obtained by projecting the triangulated landmarks on the camera plane, in red. At the same time, b), e) and c), f) show the similarity response between the rendered features and the target image dense descriptor map. The yellow color indicates a strong response, concentrated around the feature positions shown in a), demonstrating the effectiveness of our descriptor rendering approach. The small circle in blue is the circle center at the highest score response, the red circle is centered at the project landmark position.

C. Score Response

In Figure 6 we show the response score map obtained by convolving the dense descriptor map extracted using Alike-l [50] with two descriptors rendered using FaVoR $_{\text{Alike-l}}$. In particular, we draw a red circle centred at the projection of the triangulated landmarks on the camera plane. We add another circle (in blue) that is centred at the coordinates of the pixel with the strongest similarity response. Both the circles should be concentric to provide an accurate pose estimate. However, we notice that, most of the time, the circles do not have the same centre for the samples obtained from the Cambridge Landmarks dataset [17]. This misalignment may be due to an imprecise triangulation of the landmarks, given the depth uncertainty for the large scenes of the Cambridge Landmarks dataset [17].

D. Training and Losses

To train our voxel representation of the descriptor patches, we used two main losses, the squared L2 norm loss and the cosine similarity loss. We begin by training our model using the squared L2 norm and cosine similarity losses to enforce that the direction and norm of the rendered descriptors match the ground truth. The cosine similarity

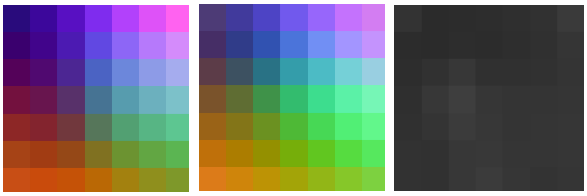


Figure 7. **Visualization of rendered vs ground truth descriptor patch.** We report the rendered descriptor patch a) and the corresponding ground-truth b) for Alike-l [50], compressed from 128 channels to 3 using PCA for visualization purposes. The patch in c) represents the normalized difference between the rendered and the ground truth, darker is better.

loss is calculated as

$$\mathrm{L}_{\cos}\left(\hat{\mathbf{d}}_{ij}^{uv}, \mathbf{d}_{ij}^{uv}\right) = 1 - \frac{\hat{\mathbf{d}}_{ij}^{uv} \cdot \mathbf{d}_{ij}^{uv}}{||\hat{\mathbf{d}}_{ij}^{uv}|| \cdot ||\mathbf{d}_{ij}^{uv}||}, \tag{7}$$

where $\hat{\mathbf{d}}_{ij}^{uv}$ is the rendered descriptor and $\mathbf{d}_{ij}^{uv}$ is the one extracted by $\mathcal{F}$ from the patch $\mathbf{P}_{ij}$. For the density grid, we use the cross-entropy loss, as in [42]. Also, to ensure a smooth representation of the descriptor patch, for the last 500 epochs, we introduce a total variation (TV) regularization term in the loss computation on the density and the de-

scriptor parameters as described in [42]. The complete loss function for the voxel optimization is given by

$$\text{Loss}\left(\hat{\mathbf{d}}_{ij}^{uv}, \mathbf{d}_{ij}^{uv}\right) = ||\hat{\mathbf{d}}_{ij}^{uv} - \mathbf{d}_{ij}^{uv}||_2^2 + \text{L}_{\cos}\left(\hat{\mathbf{d}}_{ij}^{uv}, \mathbf{d}_{ij}^{uv}\right) + \text{TV}. \quad (8)$$

During training, we choose a learning rate that depends on the visibility of each sub-voxel. In particular, we follow the same approach proposed by Sun *et al*. [42] where subvoxels visible from fewer views have a lower learning rate. Choosing the learning rate according to the visibility of the voxels allows training to *focus* more on the parts of the voxels that represent portions that have been observed from several views, and hence are more reliable than those with fewer observations. Figure 7 shows a descriptor patch, the corresponding ground truth extracted using Alike-l [50], both visualized using principal component analysis (PCA) to map the descriptor space to RGB colors, and the L2 norm between the two patches in the descriptor space.

E. Landrmark Triangulation

Our method requires landmarks positions to locate and train the associated voxels. Vision-based localization systems, such as visual-inertial odometry or visual simultaneous localization and mapping, already provide a landmark's position in 3D space. Hence, we opt for a simple-to-implement multi-view triangulation approach, since triangulation is not the main focus of our work. Given a track containing N poses $\mathbf{T}_i$, with $i = 1 \dots N$, and hence N keypoints $\mathbf{k}_{ij}$ corresponding to the projection onto each camera plane of the landmark $\boldsymbol{\ell}_j$, we wish to find the 3D coordinates ${}^{\mathcal{W}}\ell_j^x, {}^{\mathcal{W}}\ell_j^y, {}^{\mathcal{W}}\ell_j^z$ in the world frame $\mathcal{W}$ of $\boldsymbol{\ell}_j$. An initial estimate of the coordinates $\boldsymbol{\ell}_j$ can be determined by two-view triangulation methods, such as the direct linear transform. Following [10], we choose an anchor pose $\mathbf{T}_a$ from among the poses in the track, and express $\boldsymbol{\ell}_j$ in camera coordinates for $\mathbf{T}_a$. We define $\mathbf{T}_a \in \text{SE}(3)$, and in general any $\mathbf{T}_i$, in terms of a rotation matrix $\mathbf{R}_a \in \text{SO}(3)$ and a translation vector $\mathbf{t}_a \in \mathbb{R}^3$:

$$\begin{bmatrix} {}^{\mathcal{A}}\ell_j^x \\ {}^{\mathcal{A}}\ell_j^y \\ {}^{\mathcal{A}}\ell_j^z \end{bmatrix} = \mathbf{R}_a^T \begin{bmatrix} {}^{\mathcal{W}}\ell_j^x \\ {}^{\mathcal{W}}\ell_j^y \\ {}^{\mathcal{W}}\ell_j^z \end{bmatrix} - \mathbf{R}_a^T \mathbf{t}_a. \quad (9)$$

From Equation (9), we can write the landmark coordinates in the world frame as a function of the anchor pose coordinates:

$$\begin{bmatrix} {}^{\mathcal{W}}\ell_j^x \\ {}^{\mathcal{W}}\ell_j^y \\ {}^{\mathcal{W}}\ell_j^z \end{bmatrix} = \mathbf{R}_a \begin{bmatrix} {}^{\mathcal{A}}\ell_j^x \\ {}^{\mathcal{A}}\ell_j^y \\ {}^{\mathcal{A}}\ell_j^z \end{bmatrix} + \mathbf{t}_a. \quad (10)$$

Hence, each time we need to determine the landmark coordinates $\boldsymbol{\ell}_j$ in camera frame $\mathcal{T}_i$, associated with the pose $\mathbf{T}_i$ in the track, we can write:

$$\begin{bmatrix} {}^{\mathcal{T}_i}\ell_j^x \\ {}^{\mathcal{T}_i}\ell_j^y \\ {}^{\mathcal{T}_i}\ell_j^z \end{bmatrix} = \mathbf{R}_i^T \left(\mathbf{R}_a \begin{bmatrix} {}^{\mathcal{A}}\ell_j^x \\ {}^{\mathcal{A}}\ell_j^y \\ {}^{\mathcal{A}}\ell_j^z \end{bmatrix} + \mathbf{t}_a \right) - \mathbf{R}_i^T \mathbf{t}_i \quad (11)$$

$$= \mathbf{R}_i^T \mathbf{R}_a \begin{bmatrix} {}^{\mathcal{A}}\ell_j^x \\ {}^{\mathcal{A}}\ell_j^y \\ {}^{\mathcal{A}}\ell_j^z \end{bmatrix} + \mathbf{R}_i^T (\mathbf{t}_a - \mathbf{t}_i) \quad (12)$$

To improve the numerical stability of the optimization process, we represent $\left[{}^{\mathcal{A}}\ell_j^x, {}^{\mathcal{A}}\ell_j^y, {}^{\mathcal{A}}\ell_j^z\right]^T$ using the inverse depth parametrization,

$$\alpha_j = \frac{{}^{\mathcal{A}}\ell_j^x}{{}^{\mathcal{A}}\ell_j^z}, \quad \beta_j = \frac{{}^{\mathcal{A}}\ell_j^y}{{}^{\mathcal{A}}\ell_j^z}, \quad \rho_j = \frac{1}{{}^{\mathcal{A}}\ell_j^z} \quad (13)$$

We can then rewrite Equation (12) as

$$\begin{bmatrix} {}^{\mathcal{T}_i}\ell_j^x \\ {}^{\mathcal{T}_i}\ell_j^y \\ {}^{\mathcal{T}_i}\ell_j^z \end{bmatrix} = \frac{1}{\rho_j} \left(\mathbf{R}_i^T \mathbf{R}_a \begin{bmatrix} \alpha_j \\ \beta_j \\ 1 \end{bmatrix} + \rho_j \, \mathbf{R}_i^T (\mathbf{t}_a - \mathbf{t}_i) \right) \quad (14)$$

In turn, the camera measurement model is

$$\hat{\mathbf{z}}_{ij} = \frac{1}{{}^{\mathcal{T}_i}\ell_j^z} \left[{}^{\mathcal{T}_i}\ell_j^x, {}^{\mathcal{T}_i}\ell_j^y \right]^T, \quad (15)$$

where $\hat{\mathbf{z}}_{ij}$ are the normalized image plane coordinates of ${}^{\mathcal{T}_i}\ell_j$. The predicted measurement can be determined by transforming $\mathbf{k}_{ij}$ into camera coordinates to obtain $\mathbf{z}_{ij}$. This involves back-projecting the keypoint coordinates $\mathbf{k}_{ij}$ from the image plane to the camera frame, followed by normalization,

$$\begin{bmatrix} x_{\mathbf{k}_{ij}} \\ y_{\mathbf{k}_{ij}} \\ z_{\mathbf{k}_{ij}} \end{bmatrix} = \mathbf{K}^{-1} \begin{bmatrix} \mathbf{k}_{ij} \\ 1 \end{bmatrix} \quad (16)$$

$$\mathbf{z}_{ij} = \frac{1}{z_{\mathbf{k}_{ij}}} \left[x_{\mathbf{k}_{ij}}, y_{\mathbf{k}_{ij}} \right]^T, \quad (17)$$

where $\mathbf{K}$ is the intrinsic camera calibration matrix.

Finally, we find α_j, β_j, and ρ_j via Levenberg-Marquardt optimization,

$$\mathbf{e}_{ij} = \mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}, \quad (18)$$

$$u_{ij} = \sqrt{\mathbf{e}_{ij}^T \mathbf{e}_{ij}}, \quad (19)$$

$$\rho(u) = \frac{1}{2} \frac{c^2 u^2}{c^2 + u^2}, \quad (20)$$

$$\alpha_j^*, \beta_j^*, \rho_j^* = \underset{\alpha_j, \beta_j, \rho_j}{\text{argmin}} \sum_{i \in \mathcal{S}_j} \rho(u_{ij}(\mathbf{e}_{ij}(\hat{\mathbf{z}}_{ij}(\boldsymbol{\ell}_{ij}(\alpha_j, \beta_j, \rho_j))))), \quad (21)$$

where $\rho(u)$ is a robust cost function [13] parameterized by c, used to prevent outliers from negatively impacting the estimate of the landmark coordinates.

F. Robustness to Pose Initialization Error

In Table 6 we report the 6-DoF median localization errors for the 7-Scenes [40] dataset using two pose initialization methods: the first frame of the test sequence (Constant) and DenseVLAD [43] (Retrieval). We perform the evaluation using FaVoR coupled with Alike-l [50]. The 'first frame' initialization choice is equivalent to adding increased noise to the starting guess, with increasing error as the target pose moves far away from the initial pose (at the first frame). However, this approach does ensure reproducibility and provides a consistent baseline for fair comparisons with future work, offering a reliable measure of our method's robustness. Our results indicate that after three iterations of the Render+PnP-RANSAC paradigm, our method converges to a low localization error, even when starting from less accurate poses than those provided by DenseVLAD [43].

Scene	Method	Prior Err.	Iter 1	Iter 2	Iter 3
chess	Retrieval	21.9, 12.13	0.8, 0.21	0.7, 0.19	0.7, 0.18
	Constant	147.6, 29.94	1.0, 0.28	0.7, 0.19	0.7, 0.18
fire	Retrieval	34.4, 13.2	0.8, 0.3	0.9, 0.4	0.8, 0.3
	Constant	96.6, 35.6	1.2, 0.5	0.9, 0.4	0.8, 0.3
heads	Retrieval	15.8, 15.0	0.7, 0.5	0.7, 0.4	0.7, 0.4
	Constant	45.7, 37.8	28.7, 17.2	2.5, 1.5	0.5, 0.4
office	Retrieval	28.6, 11.1	1.7, 0.4	1.7, 0.4	1.6, 0.4
	Constant	113.8, 67.4	158.3, 57.4	10.2, 2.5	1.7, 0.4
pumpkin	Retrieval	31.4, 10.8	1.4, 0.3	1.4, 0.3	1.3, 0.3
	Constant	137.0, 49.7	4.1, 1.1	1.5, 0.3	1.2, 0.2
redkitchen	Retrieval	29.4, 12.0	1.2, 0.3	1.2, 0.3	1.2, 0.2
	Constant	192.4, 39.3	7.8, 1.8	1.8, 0.4	1.1, 0.2
stairs	Retrieval	26.2, 15.8	3.8, 1.0	3.0, 0.8	2.7, 0.8
	Constant	178.9, 16.3	231.4, 34.2	106.8, 16.2	4.7, 1.32

Table 6. **Different pose initialization priors for 7-Scenes dataset [40]**. We report the 6-DoF median pose errors (cm, deg) obtained with FaVoR $_{\text{Alike-l}}$ for different pose initialization methods. The results show that FaVoR is robust to different initial poses as it converges to small localization errors after iterating the Render+PnP-RANSAC scheme.

Scene	Iteration	Feature Extractor	Initial pose error (cm, deg)	Estimated pose error (cm, deg)	Avg. N. of inliers	Success rate (%)
chess	1st	alike-l	21.92, 12.13	0.77, 0.21	66	100.00
	2nd		-	0.74, 0.19	74	100.00
	3rd		-	0.72, 0.18	74	100.00
	1st	alike-n	21.92, 12.13	0.73, 0.21	64	100.00
	2nd		-	0.68, 0.18	71	100.00
	3rd		-	0.64, 0.17	71	100.00
	1st	alike-s	21.92, 12.13	0.82, 0.26	117	100.00
	2nd		-	0.76, 0.22	122	100.00
	3rd		-	0.71, 0.20	122	100.00
	1st	alike-t	21.92, 12.13	1.01, 0.29	66	100.00
	2nd		-	0.94, 0.27	73	100.00
	3rd		-	0.89, 0.25	73	100.00
	1st	SuperPoint	0.22, 12.13	0.68, 0.19	88	100.00
	2nd		-	0.67, 0.17	99	100.00
	3rd		-	0.64, 0.16	99	100.00
fire	1st	alike-l	34.37, 13.23	0.82, 0.34	73	100.00
	2nd		-	0.88, 0.36	72	100.00
	3rd		-	0.83, 0.34	72	99.85
	1st	alike-n	34.37, 13.23	0.86, 0.37	66	100.00
	2nd		-	0.93, 0.37	66	100.00
	3rd		-	0.89, 0.35	66	99.60
	1st	alike-s	34.37, 13.23	1.22, 0.47	172	100.00
	2nd		-	1.64, 0.60	162	100.00
	3rd		-	1.52, 0.56	163	99.70
	1st	alike-t	34.37, 13.23	1.17, 0.47	59	100.00
	2nd		-	1.37, 0.52	55	100.00
	3rd		-	1.30, 0.49	55	99.55
	1st	SuperPoint	34.37, 13.23	1.02, 0.39	67	100.00
	2nd		-	1.04, 0.38	67	100.00
	3rd		-	0.98, 0.36	67	98.65
heads	1st	alike-l	15.77, 14.97	0.73, 0.46	46	100.00
	2nd		-	0.67, 0.41	53	100.00
	3rd		-	0.66, 0.40	53	94.50
	1st	alike-n	15.77, 14.97	1.08, 0.59	38	100.00
	2nd		-	0.97, 0.53	43	100.00
	3rd		-	0.96, 0.56	43	91.30
	1st	alike-s	15.77, 14.97	0.70, 0.43	81	100.00
	2nd		-	0.62, 0.37	92	100.00
	3rd		-	0.59, 0.36	92	99.20
	1st	alike-t	15.77, 14.97	0.89, 0.52	52	100.00
	2nd		-	0.81, 0.48	59	100.00
	3rd		-	0.76, 0.44	59	98.90
	1st	SuperPoint	15.77, 14.97	0.62, 0.39	76	100.00
	2nd		-	0.54, 0.34	87	100.00
	3rd		-	0.52, 0.32	88	99.20

Continue on next page

office	1st	alike-l	28.58, 11.06	1.69, 0.43	36	100.00
	2nd		-	1.68, 0.41	39	100.00
	3rd		-	1.63, 0.39	40	99.25
	1st	alike-n	28.58, 11.06	1.77, 0.47	35	100.00
	2nd		-	1.74, 0.45	37	100.00
	3rd		-	1.69, 0.42	37	97.78
	1st	alike-s	28.58, 11.06	1.63, 0.45	69	100.00
	2nd		-	1.56, 0.41	72	100.00
	3rd		-	1.55, 0.40	72	99.98
	1st	alike-t	28.58, 11.06	2.57, 0.68	37	100.00
	2nd		-	2.26, 0.61	42	100.00
	3rd		-	2.21, 0.58	42	99.15
	1st	SuperPoint	28.58, 11.06	1.75, 0.43	65	100.00
	2nd		-	1.71, 0.41	72	100.00
	3rd		-	1.64, 0.37	72	99.75
pumpkin	1st	alike-l	31.38, 10.81	1.39, 0.30	69	100.00
	2nd		-	1.39, 0.29	76	100.00
	3rd		-	1.31, 0.28	76	98.65
	1st	alike-n	31.38, 10.81	1.58, 0.36	70	100.00
	2nd		-	1.53, 0.34	76	100.00
	3rd		-	1.46, 0.31	76	93.45
	1st	alike-s	31.38, 10.81	1.38, 0.31	118	100.00
	2nd		-	1.38, 0.29	121	100.00
	3rd		-	1.34, 0.28	122	99.25
	1st	alike-t	31.38, 10.81	1.87, 0.43	81	100.00
	2nd		-	1.70, 0.39	90	100.00
	3rd		-	1.67, 0.37	91	96.40
	1st	SuperPoint	31.38, 10.81	1.50, 0.33	110	100.00
	2nd		-	1.51, 0.31	120	100.00
	3rd		-	1.45, 0.29	120	99.05
redkitchen	1st	alike-l	29.38, 11.97	1.23, 0.30	45	100.00
	2nd		-	1.18, 0.25	54	100.00
	3rd		-	1.15, 0.24	54	98.08
	1st	alike-n	29.38, 11.97	1.37, 0.33	51	100.00
	2nd		-	1.34, 0.32	60	100.00
	3rd		-	1.21, 0.28	60	96.02
	1st	alike-s	29.38, 11.97	4.66, 1.09	57	100.00
	2nd		-	4.28, 1.00	67	100.00
	3rd		-	4.03, 0.94	68	77.42
	1st	alike-t	29.38, 11.97	1.44, 0.31	57	100.00
	2nd		-	1.39, 0.29	66	100.00
	3rd		-	1.33, 0.27	67	99.38
	1st	SuperPoint	0.29, 11.97	1.38, 0.30	79	100.00
	2nd		-	1.42, 0.27	93	100.00
	3rd		-	1.33, 0.24	93	98.74

Continue on next page

stairs	1st	alike-l	26.19, 15.81	3.80, 1.03	11	100.00
	2nd		-	3.02, 0.81	12	100.00
	3rd		-	2.74, 0.82	12	97.90
	1st	alike-n	26.19, 15.81	7.19, 1.97	10	100.00
	2nd		-	6.28, 1.65	10	100.00
	3rd		-	5.96, 1.59	10	93.10
	1st	alike-s	26.19, 15.81	5.78, 1.63	70	100.00
	2nd		-	5.18, 1.49	68	100.00
	3rd		-	5.03, 1.51	68	100.00
	1st	alike-t	26.19, 15.81	5.30, 1.49	14	100.00
	2nd		-	4.38, 1.20	15	100.00
	3rd		-	4.03, 1.07	15	100.00
	1st	SuperPoint	26.19, 15.81	5.83, 1.69	27	100.00
	2nd		-	4.54, 1.21	31	100.00
	3rd		-	4.05, 1.07	31	99.90
Overall Average	1st	alike-t	26.80, 12.85	2.03, 0.60	52	99.89
	2nd		-	1.83, 0.54	57	99.24
	3rd		-	1.74, 0.50	57	99.05
	1st	alike-s	26.80, 12.85	2.31, 0.66	97	97.83
	2nd		-	2.20, 0.63	100	96.83
	3rd		-	2.11, 0.61	101	96.51
	1st	alike-n	26.80, 12.85	2.08, 0.62	47	97.18
	2nd		-	1.93, 0.55	51	96.28
	3rd		-	1.83, 0.52	51	95.89
	1st	alike-l	26.80, 12.85	1.49, 0.44	49	99.25
	2nd		-	1.37, 0.39	54	98.65
	3rd		-	1.29, 0.38	54	98.32
	1st	SuperPoint	26.80, 12.85	1.82, 0.53	73	99.81
	2nd		-	1.63, 0.44	81	99.49
	3rd		-	1.52, 0.40	81	99.33

Table 7. **6-DoF median localization errors on the 7-Scenes dataset [40]** for the various features extractors used to train FaVoR.

Scene	Iteration	Feature Extractor	Initial pose error (cm, deg)	Estimated pose error (cm, deg)	Avg. N. of inliers	Success rate (%)
Shop	1st	alike-l	136.31, 7.19	5.38, 0.27	208	100.00
	2nd		-	5.63, 0.22	231	100.00
	3rd		-	5.48, 0.25	231	100.00
	1st	alike-n	136.31, 7.19	5.27, 0.28	185	100.00
	2nd		-	5.43, 0.23	208	100.00
	3rd		-	5.09, 0.24	208	100.00
	1st	alike-s	136.31, 7.19	5.99, 0.24	225	100.00
	2nd		-	5.76, 0.25	250	100.00
	3rd		-	6.05, 0.25	250	100.00
	1st	alike-t	136.31, 7.19	5.65, 0.27	203	100.00
	2nd		-	5.89, 0.26	224	100.00
	3rd		-	5.25, 0.25	224	100.00
	1st	SuperPoint	136.31, 7.19	5.87, 0.29	204	100.00
	2nd		-	5.20, 0.26	225	100.00
	3rd		-	5.47, 0.26	224	100.00
College	1st	alike-l	289.98, 5.96	18.19, 0.25	359	100.00
	2nd		-	17.04, 0.26	373	100.00
	3rd		-	15.25, 0.23	372	100.00
	1st	alike-n	289.98, 5.96	16.82, 0.28	315	100.00
	2nd		-	17.38, 0.28	327	100.00
	3rd		-	17.61, 0.26	327	100.00
	1st	alike-s	289.98, 5.96	16.64, 0.27	327	100.00
	2nd		-	15.74, 0.26	338	100.00
	3rd		-	15.67, 0.24	338	100.00
	1st	alike-t	289.98, 5.96	17.64, 0.28	326	100.00
	2nd		-	16.33, 0.26	336	100.00
	3rd		-	16.52, 0.25	337	100.00
	1st	superpoint	289.98, 5.96	17.88, 0.27	326	100.00
	2nd		-	18.15, 0.28	336	100.00
	3rd		-	17.52, 0.27	336	100.00
Great	1st	alike-l	719.21, 9.47	32.46, 0.16	103	100.00
	2nd		-	29.48, 0.15	116	100.00
	3rd		-	27.40, 0.14	116	99.87
	1st	alike-n	719.21, 9.47	37.88, 0.21	82	100.00
	2nd		-	35.20, 0.18	91	100.00
	3rd		-	32.05, 0.18	91	99.21
	1st	alike-s	719.21, 9.47	36.18, 0.19	94	100.00
	2nd		-	34.27, 0.18	105	100.00
	3rd		-	31.78, 0.16	106	99.87
	1st	alike-t	719.21, 9.47	33.78, 0.19	101	100.00
	2nd		-	31.33, 0.15	114	100.00
	3rd		-	28.83, 0.14	114	100.00
	1st	SuperPoint	719.21, 9.47	34.69, 0.22	142	100.00
	2nd		-	30.71, 0.20	161	100.00
	3rd		-	29.09, 0.20	161	100.00

Continue on next page

Hospital	1st	alike-l	405.22, 7.58	22.18, 0.44	155	100.00
	2nd		-	21.37, 0.40	160	100.00
	3rd		-	19.37, 0.36	160	100.00
	1st	alike-n	405.22, 7.58	27.28, 0.47	128	100.00
	2nd		-	22.68, 0.44	132	100.00
	3rd		-	21.17, 0.40	131	100.00
	1st	alike-s	405.22, 7.58	25.13, 0.44	132	100.00
	2nd		-	25.71, 0.47	136	100.00
	3rd		-	20.75, 0.37	136	100.00
	1st	alike-t	405.22, 7.58	26.30, 0.51	140	100.00
	2nd		-	25.10, 0.48	145	100.00
	3rd		-	20.14, 0.41	145	100.00
	1st	SuperPoint	405.22, 7.58	31.53, 0.56	143	100.00
	2nd		-	31.05, 0.55	148	100.00
	3rd		-	27.46, 0.54	148	100.00
Church	1st	alike-l	287.61, 9.36	11.58, 0.38	201	100.00
	2nd		-	10.31, 0.31	228	100.00
	3rd		-	10.35, 0.30	228	100.00
	1st	alike-n	287.61, 9.36	12.46, 0.43	181	100.00
	2nd		-	11.53, 0.35	206	100.00
	3rd		-	10.90, 0.33	206	100.00
	1st	alike-s	287.61, 9.36	12.67, 0.42	180	100.00
	2nd		-	12.01, 0.36	203	100.00
	3rd		-	11.40, 0.35	204	99.81
	1st	alike-t	287.61, 9.36	12.18, 0.40	170	100.00
	2nd		-	11.66, 0.35	192	100.00
	3rd		-	11.21, 0.36	192	100.00
	1st	SuperPoint	287.61, 9.36	14.15, 0.49	188	100.00
	2nd		-	12.72, 0.42	220	100.00
	3rd		-	11.43, 0.38	220	99.81
Overall Average	1st	alike-t	367.67, 7.91	19.11, 0.33	188	100.00
	2nd		-	18.06, 0.30	202	100.00
	3rd		-	16.39, 0.28	202	100.00
	1st	alike-s	367.67, 7.91	19.32, 0.31	192	100.00
	2nd		-	18.70, 0.30	206	100.00
	3rd		-	17.13, 0.27	207	99.94
	1st	alike-n	367.67, 7.91	19.94, 0.34	178	100.00
	2nd		-	18.44, 0.30	193	99.92
	3rd		-	17.36, 0.28	193	99.84
	1st	alike-l	367.67, 7.91	17.96, 0.30	205	100.00
	2nd		-	16.77, 0.27	222	100.00
	3rd		-	15.57, 0.26	221	99.97
	1st	SuperPoint	367.67, 7.91	20.82, 0.37	201	100.00
	2nd		-	19.57, 0.34	218	99.96
	3rd		-	18.19, 0.33	218	99.96

Table 8. **6-DoF median localization errors on the Cambridge dataset [17]** for the various features extractors used to train FaVoR.